网络硬盘录像机
客户端软件开发包编程手册

版本0.0.3

16-06-2009

DVR windows sdk

一、历史

2008-11-20	V0.0.1	朱履斌	完成 SDK 第一版本
2009-03-21	V0.0.2	陈杰	支持 R8016
2009-06-01	V0.0.3	陈杰	支持 R9000

二、 目标

提供给客户二次开发环境所需的头文件和库文件。

三、框架：

客户端软件开发包是网络视频服务器的配套产品。

开发包中所实现的各函数功能主要油客户端处理和主机端控制两个部分组成，本手册详细介绍了此软件开发包中各结构体、宏、枚举的定义（详见六）及各函数的定义、使用方法及相互调用关系（详见七）。

四、 关键字定义：

函数名、枚举、结构体、宏定义以 NETDVR_开头。

五、 SDK APIs 返回值定义 (有待补充)

```
enum NETDVR_RETURN_CODE
{
    NETDVR_SUCCESS = 0,                     //成功
    NETDVR_REC_STATUS_STARTED,              //未使用
    NETDVR_REC_STATUS_STOPPED,              //未使用

    NETDVR_ERROR = 2000,                    //一般错误
    NETDVR_ERR_UNKNOWN,                     //未知错误

    NETDVR_ERR_PARAM,                       //参数输入错误
    NETDVR_ERR_RET_PARAM,                   //命令返回参数错误
    NETDVR_ERR_NOINIT,                      //DVR未创建

    NETDVR_ERR_COMMAND,                     //命令错误

    NETDVR_ERR_NEW_CONNECTION,              //连接建立失败
    NETDVR_ERR_SEND,                        //命令发送失败

    NETDVR_ERR_OUTOFMEMORY,                 //内存越界或不足
    NETDVR_ERR_RESOURCE,                    //资源错误
    NETDVR_ERR_FILENOTEXIST,                //文件不存在
    NETDVR_ERR_BAUDLIMIT,                   //每一路通道最多支持5路实时监控
    NETDVR_ERR_CREATESOCKET,                //创建套结字失败
    NETDVR_ERR_SOCKET,                      //网络socket错误
    NETDVR_ERR_CONNECT,                     //网络连接失败
```

```
    NETDVR_ERR_BIND,                    //绑定失败
    NETDVR_ERR_LISTEN,                  //端口监听错误
    NETDVR_ERR_NETSND,                  //网络发送出错
    NETDVR_ERR_NETRCV,                  //网络接收出错
    NETDVR_ERR_TIMEOUT,                 //网络连接超时
    NETDVR_ERR_CHNERROR,                //超出通道限制
    NETDVR_ERR_DEVICEBUSY,              //设备正在忙
    NETDVR_ERR_WRITEFLASH,              //烧写flash出错
    NETDVR_ERR_VERIFY,                  //校验错
    NETDVR_ERR_CONFLICT,                //系统资源冲突
    NETDVR_ERR_BUSY,                    //系统正在忙
    NETDVR_ERR_USER_SAME,               //用户名重复
    NETDVR_ERR_LINKLIMIT,               //连接数限制
    NETDVR_ERR_DATABASE,                //错误数据库

    /* === error code for login === */
    NETDVR_ERR_NOUSER,                  //用户不存在
    NETDVR_ERR_PASSWD,                  //密码错误
    NETDVR_ERR_MACADDR,                 //MAC地址错误
    NETDVR_ERR_RELOGIN,                 //重复登录错误
    NETDVR_ERR_NOLOGIN,                 //用户未登录操作失败

    /* === net player === */
    NETDVR_ERR_NET_PLAYER_FULL,         //播放连接已满

    /* === updateing ==== */
    NETDVR_ERR_UPDATING,                //升级错误

    /* === remote file download error === */
    NETDVR_ERR_FILEOPEN,                //文件下载打开失败
    NETDVR_ERR_USERSTOPPED,             //文件下载用户停止

    NETDVR_ERR_SERIAL_REOPEN,           //未使用

    NETDVR_ERR_GET_LOCALMACADDR,        //获得本地mac地址错误

    NETDVR_ERR_SDK_CHECKFAILED          //校验SDK失败
};
```

六、 结构体、宏、枚举及函数指针定义

1、 DVR 服务器信息

```
enum NETDVR_SERVERTYPE
```

```
    {
        SERVER_R6104,
        SERVER_R8016,
        SERVER_R9000,

    };

struct NETDVR_serverInfo_t
{
    unsigned int serverip;                          //服务器ip，必要  /*IN*/
    unsigned short serverport;                          //服务器控制端口，必要/*IN*//*OUT*/
    char servername[30];                            //DVR名称，非必要
    unsigned char main_chn_num;                         //主码流通道数，必要/*IN*//*OUT*/
    unsigned char sub_chn_num;                      //子码流通道数，必要/*IN*//*OUT*/
    unsigned char alarm_in_num;                     //报警输入通道数/*IN*//*OUT*/
    unsigned char alarm_out_num;                        //报警输出通道数/*IN*//*OUT*/
    NETDVR_SERVERTYPE servertype;       //必要
    unsigned char reserved[60];             //reserved for extensible development
};
```

2、连接断开回调函数指针

```
typedef void (*PFUN_MSG_T)(unsigned int dwContent);
```

3、登录信息

```
struct NETDVR_loginInfo_t
{
    char username[12];                              //登录名，必要
    char loginpass[12];                             //登录密码，必要
    char macAddr[18];                               //客户端 mac 地址，必要，空则使用默认
    unsigned int ipAddr;                            //客户端 ip，必要，0 则自动获得
};
```

4、密码修改信息

```
struct NETDVR_loginPass_t
{
    char name[12];                                  //需要修改密码那的用户名称
    char oldpass[12];                               //旧密码
    char newpass[12];                               //新密码
    char confirm[12];                               //新密码确认
};
```

5、码流接收设置信息

```
struct NETDVR_mediaRcvParam_t
{
```

```
    unsigned int ip;                                    //码流接收 ip（0 自动获得本机 ip）
    unsigned short port;                                //码流接收端口（0 自动获得端口）
    unsigned char chn;                                  //码流通道
    char reserved[5];                                   //保留字段
};
```

6、原始码流接收回调函数指针

```
typedef void (* pFrameCallBack)(pFrameHeadr pFrmHdr, unsigned int dwContext);
```

pFrameHeadr：参考六-7

7、原始码流头信息

```
typedef struct FrameHeadr
{
    unsigned char mediaType;                            //码流类型
    unsigned char *pData;                               //码流数据
    unsigned int preBufSize;                            //未使用
    unsigned int dataSize;                              //码流数据长度
    unsigned char frameRate;                            //帧率
    unsigned int frameID;                               //帧 id
    unsigned int timeStamp;                             //时间戳
    union
    {
        struct{
            int keyFrame;                               //是否视频关键帧
            unsigned short videoWidth;                  //视频帧宽
            unsigned short videoHeight;                 //视频帧高
        } videoParam;
        unsigned char audioMode;                        //音频采样位数 8, 16, or 32 bit
    };
} FrameHeadr;
typedef FrameHeadr* pFrameHeadr;                        //定义原始码流头指针类型
```

8、视频解码格式

```
typedef enum NETDVR_FMT_TYPE {
    NETDVR_FMT_RGB24 = 2,                               //rgb24
    NETDVR_FMT_RGB32 = 4,                               //rgb32
    NETDVR_FMT_YV12 = 6,                                //yv12
    NETDVR_FMT_I420 = 8,                                //i420
    NETDVR_FMT_YUY2 = 10,                               //yuy2(暂不支持抓图)
} fmt_type_t;
```

注,R8016 只支持 NETDVR_FMT_YV12 格式

9、解码数据回调函数

```
typedef void (* pDecFrameCallBack)(pFrameHeadrDec pFrmHdrDec, unsigned int dwContext);
```
pFrmHdrDec：参考六-10

10、解码后码流头信息

```
typedef struct FrameHeadrDec
{
    unsigned char mediaType;                    //解码前码流类型:
    char reserved1[3];                          //保留
    void *data;                                 //解码后数据
    unsigned int data_size;                     //解码后数据长度
    char reserved2[32];                         //保留
    union
    {
        struct{
            fmt_type_t fmt;                     //解码后视频帧格式
            unsigned short width;               //视频宽
            unsigned short height;              //视频高
        } video_param;
        unsigned char audio_mode;               //音频采样位数 8, 16, or 32 bit
    };
} FrameHeadrDec;
typedef FrameHeadrDec* pFrameHeadrDec;          //定义解码后码流头指针类型
```

11、录像文件检索条件

```
struct NETDVR_fileSearchCondition_t
{
    int chn;                                    //检索通道
    unsigned char type;                         //检索录像文件类型（参阅六-12）
    unsigned int start_time;                    //检索录像开始时间（time_t 类型）
    unsigned int end_time;                      //检索录像结束时间（time_t 类型）
    unsigned short startID;                     //从检索结果第 startID 条开始返回 (>=1)
    unsigned short max_return;                  //最多一次返回的录像文件数 (<=24)
};
```

12、录像文件检索类型

```
enum NETDVR_REC_INDEX_MASK {
    NETDVR_REC_INDEX_TIMER = 0x1,               //定时录像
    NETDVR_REC_INDEX_MD = 0x2,                  //移动侦测录像
    NETDVR_REC_INDEX_ALARM = 0x4,               //报警录像
    NETDVR_REC_INDEX_MANUAL = 0x8,              //手动录像
    NETDVR_REC_INDEX_ALL = 0x10,
};
```

13、录像文件检索结果

```
struct NETDVR_recFileSearchResult_t
{
    unsigned short sum;                         //检索到的总文件数
    unsigned short startID;                     //从第startID个开始返回，0则无记录
    unsigned short endID;                       //返回第endID个文件结束
    struct NETDVR_recFileInfo_t *precInfo;      //返回文件队列头指针（参阅六-14）
};
```

14、录像文件信息

```
struct NETDVR_recFileInfo_t
{
    unsigned char channel_no;                   //录像通道
    unsigned char type;                         //录像类型
    unsigned int start_time;                    //录像开始时间
    unsigned int end_time;                      //录像结束时间
    unsigned char image_format;                 //帧格式:3-cif ; 4-4cif
    unsigned char stream_flag;                  //流类型:0-视频 ; 1 – 复合
    unsigned int size;                          //文件大小
    unsigned int offset;                        //相对文件池的偏移量
    char filename[64];                          //文件名
    struct NETDVR_recFileInfo_t *pnext;         //队列指针（指向下一个录像文件）
};
```

15、定义进度回调函数指针类型

```
typedef void (*PFUN_PROGRESS_T)(struct NETDVR_progressParam_t progress, unsigned int dwContent);
```

progress：参阅六-16

16、进度结构体

```
struct NETDVR_progressParam_t
{
    unsigned int curr_pos;                      //当前完成大小
    unsigned int total_size;                    //需完成大小
};
```

17、定义下载接收线程错误消息回调函数指针类型

```
typedef void (*PFUN_ERROR_T)(unsigned short err_code, unsigned int dwContent);
```

err_code：参阅五

18、回放接收设置

```
struct NETDVR_playerRcvPara_t
{
    unsigned int ip;                            //码流接收ip（0自动获得本机ip）
    unsigned short video_port;                  //视频码流接收端口（0自动获得端口）
    unsigned short audio_port;                  //视频码流接收端口（0自动获得端口）
```

```
    unsigned char player_index;                 //回放设备号
    char reserved[7];                           //保留
};
```

19、定义回放结束消息回调函数指针类型

```
typedef void (*PFUN_MSG_T)(unsigned int dwContent);
```

20、定义回放文件是否含音频消息回调函数指针类型

```
typedef void (*PFUN_MSGHASAUDIO_T)(unsigned char b_has_audio, unsigned int dwContent);
```

21、系统参数

```
struct NETDVR_systemParam_t
{
    unsigned short device_id;                   //设备id
    char device_name[32];                       //设备名称
    unsigned int    time;                       //系统时间
    enum NETDVR_OVERLAP flag_overlap;           //硬盘满时是否重新覆盖（参阅六-22）
    enum NETDVR_DVRSTATUS flag_statusdisp;//是否显示系统状态（参阅六-23）
    enum NETDVR_LOCKTIME lock_time;             //锁定时间（参阅六-24）
    enum NETDVR_SWITCHTIME switch_time;         //自动切换时间（参阅六-25）
    enum NETDVR_VIDEOFORMAT video_format;//PAL/ NTSC制（参阅六-26）
    enum NETDVR_VGARESOLUTION vga_solution;//vga分辨率（参阅六-27）
    enum NETDVR_TRANSPARENCY transparency;//菜单透明度（参阅六-28）
    enum NETDVR_LANGUAGE language;              //语种（参阅六-29）

    char reserved[32];                          //保留
};
```

22、硬盘满是否覆盖

```
enum NETDVR_OVERLAP {
    NETDVR_OVERLAP_NO = 0,                      //硬盘满不覆盖
    NETDVR_OVERLAP_YES = 1,                     //硬盘满覆盖
};
```

23、是否显示系统状态

```
enum NETDVR_DVRSTATUS {
    NETDVR_DVRSTATUS_HIDDEN = 0,                //不显示系统状态
    NETDVR_DVRSTATUS_DISPLAY = 1,               //显示系统状态
};
```

24、系统锁定时间

```
enum NETDVR_LOCKTIME {
    NETDVR_LOCKTIME_NEVER = 0,                  //从不
    NETDVR_LOCKTIME_MIN_1 = 1,                  //1 minute
```

```
    NETDVR_LOCKTIME_MIN_2 = 2,          //2 minutes
    NETDVR_LOCKTIME_MIN_5 = 3,          //5 minutes
    NETDVR_LOCKTIME_MIN_10 = 4,         //10 minutes
    NETDVR_LOCKTIME_MIN_20 = 5,         //20 minutes
    NETDVR_LOCKTIME_MIN_30 = 6,         //30 minutes
};
```

25、系统画面自动切换时间

```
enum NETDVR_SWITCHTIME {
    NETDVR_SWITCHTIME_NEVER = 0,        //从不
    NETDVR_SWITCHTIME_SEC_5 = 1,    //5s
    NETDVR_SWITCHTIME_SEC_10 = 2,       //10s
    NETDVR_SWITCHTIME_SEC_20 = 3,       //20s
    NETDVR_SWITCHTIME_SEC_30 = 4,       //30s
    NETDVR_SWITCHTIME_MIN_1 = 5,        //1min
    NETDVR_SWITCHTIME_MIN_2 = 6,        //2min
    NETDVR_SWITCHTIME_MIN_5 = 7,        //5min
};
```

26、视频制式

```
enum NETDVR_VIDEOFORMAT {
    NETDVR_VIDEOFORMAT_PAL = 0,         //PAL制
    NETDVR_VIDEOFORMAT_NTSC = 1,        //NTSC制
};
typedef enum NETDVR_VIDEOFORMAT video_format_t;
```

27、vga分辨率

```
enum NETDVR_VGARESOLUTION {
    NETDVR_VGARESOL_1280X1024_60HZ = 0,
    NETDVR_VGARESOL_1024X768_75HZ = 1,
    NETDVR_VGARESOL_1024X768_60HZ = 2,
    NETDVR_VGARESOL_800X600_75HZ = 3,
    NETDVR_VGARESOL_800X600_60HZ = 4,
};
typedef enum NETDVR_VGARESOLUTION vga_resolution_t;
```

28、菜单透明度

```
enum NETDVR_TRANSPARENCY {
    NETDVR_TRANSPARENCY_NO = 0,         //不透明
    NETDVR_TRANSPARENCY_LOW = 1,        //低透明度
    NETDVR_TRANSPARENCY_MID = 2,        //中透明度
    NETDVR_TRANSPARENCY_HIGH = 3,       //高透明度
};
```

29、语种

```
enum NETDVR_LANGUAGE {
    NETDVR_LANGUAGE_SC = 0,                 //简体中文
    NETDVR_LANGUAGE_EN = 1,                 //英语
};
typedef enum NETDVR_LANGUAGE language_t;
```

30、云台控制参数

```
struct NETDVR_ptzParam_t
{
    unsigned short address;                 //硬件地址
    enum NETDVR_BAUDRATE baud_ratesel;      //参阅六-31
    enum NETDVR_DATABITSEL data_bitsel;     //参阅六-32
    enum NETDVR_STOPBITSEL stop_bitsel; //参阅六-33
    enum NETDVR_CHECK_TYPE check_type;      //参阅六-34
    enum NETDVR_FLOWCONTROL flow_control; //参阅六-35
    enum NETDVR_PROTOCOLTYPE protocol; //参阅六-36
    char reserved[32];
};
```

31、波特率

```
enum NETDVR_BAUDRATE {
    NETDVR_BAUDRATE_115200 = 0,
    NETDVR_BAUDRATE_57600,
    NETDVR_BAUDRATE_38400,
    NETDVR_BAUDRATE_19200,
    NETDVR_BAUDRATE_9600,
    NETDVR_BAUDRATE_4800,
    NETDVR_BAUDRATE_2400,
    NETDVR_BAUDRATE_1200,
    NETDVR_BAUDRATE_300,
};
```

32、数据位

```
enum NETDVR_DATABITSEL {
    NETDVR_DATABITSEL_8 = 0,                //8位
    NETDVR_DATABITSEL_7,                    //7位
    NETDVR_DATABITSEL_6,                    //6位
};
```

33、停止位

```
enum NETDVR_STOPBITSEL {
    NETDVR_STOPBITSEL_1 = 0,                // 1位
    NETDVR_STOPBITSEL_2,                    //2位
```

```
};
```

34、校验

```
enum NETDVR_CRCCHECK {
    NETDVR_CRCCHECK_NO = 0,                 //无校验
    NETDVR_CRCCHECK_ODD,                    //奇校验
    NETDVR_CRCCHECK_EVEN,                   //偶校验
};
```

35、流控

```
enum NETDVR_FLOWCONTROL {
    NETDVR_FLOWCONTROL_NO = 0,      //无控制
    NETDVR_FLOWCONTROL_HARDWARE,    //硬件控制
    NETDVR_FLOWCONTROL_XON_XOFF,    //Xon/Xoff
};
```

36、协议

```
enum NETDVR_PROTOCOLTYPE {
    NETDVR_PROTOCOLTYPE_PELCO_D = 0,    //pelco-d
    NETDVR_PROTOCOLTYPE_PELCO_P,        //pelco-p
    NETDVR_PROTOCOLTYPE_B01,            //b01
    NETDVR_PROTOCOLTYPE_SAMSUNG ,       //Samsung
};
```

37、云台控制命令

```
enum NETDVR_PTZCONTROL {
    NETDVR_PTZ_MOVE_UP,                 //向上转
    NETDVR_PTZ_MOVE_DOWN,               //向下转
    NETDVR_PTZ_MOVE_LEFT,               //向左转
    NETDVR_PTZ_MOVE_RIGHT,              //向右转
    NETDVR_PTZ_APER_INCREASE,           //光圈放大
    NETDVR_PTZ_APER_DECREASE,           //光圈缩小
    NETDVR_PTZ_FOCUS_FARTHER,           //改变焦距：拉远
    NETDVR_PTZ_FOCUS_NEARER,            //改变焦距：拉近
    NETDVR_PTZ_ZOOM_IN,                 //放大
    NETDVR_PTZ_ZOOM_OUT,                //缩小
    NETDVR_PTZ_LIGHT_ON,                    //灯光打开
    NETDVR_PTZ_LIGHT_OFF,               //灯光关闭
    NETDVR_PTZ_WIPER_ON,                    //雨刷打开
    NETDVR_PTZ_WIPER_OFF,               //雨刷关闭
};
```

38、巡航宏定义

```
#define MAX_PRESET_NUM 128              //预制点最大值
```

```
#define MAX_CRUISE_PATH_NUM 16                  //巡航路径号最大值
#define MAX_CRUISE_POS_NUM 16                   //每条巡航路径最大预制点数
#define MAX_CRUISE_SPEED 9                      //最大巡航速度
#define MAX_DWELL_TIME 255                      //每个巡航点最多滞留时间
```

39、巡航路径

```
struct NETDVR_cruisePath_t
{
    unsigned char path_no;                      //巡航路径号（参阅六-38）
    struct NETDVR_cruisePos_t cruise_pos[MAX_CRUISE_POS_NUM]; //巡航点
};
```

MAX_CRUISE_POS_NUM：（参阅六-38）

40、巡航点

```
struct NETDVR_cruisePos_t
{
    unsigned char cruise_no;                    //巡航点号（参阅六-38）
    unsigned char preset_no;                    //预制点号（参阅六-38）
    unsigned char dwell_time;                   //滞留时间（参阅六-38）
    unsigned char cruise_speed;                 //巡航速度（参阅六-38）
};
```

41、log检索条件

```
struct NETDVR_logSearchCondition_t
{
    logsearch_mode_t query_mode;                //检索方式（参阅六-42）
    logsearch_main_t main_type;                 //检索log主类型（参阅六-43）
    logsearch_slave_t slave_type;               //子类型（参阅六-44）
    unsigned short max_return;                  //最大返回记录数 (<=12)
    unsigned short startID;                     //从第startID条开始返回 (>=1)
    unsigned int start_time;                    //检索日志开始时间
    unsigned int end_time;                      //检索日志结束时间
    language_t langid;                          //语种（参阅六-29）
};
```

42、检索方式

```
typedef enum NETDVR_LOGSEARCH_MODE {
    NETDVR_LOGSEARCH_MODE_BOTH = 0,   //按类型和时间
    NETDVR_LOGSEARCH_MODE_TYPE,       //按类型
    NETDVR_LOGSEARCH_MODE_TIME,       //按时间
} logsearch_mode_t;
```

43、主类型

```
typedef enum NETDVR_LOGSEARCH_MAIN {
```

```
    NETDVR_LOGSEARCH_MAIN_ALARM = 0,//报警类
    NETDVR_LOGSEARCH_MAIN_LOCALOP,   //本地操作类
    NETDVR_LOGSEARCH_MAIN_REMOTEOP,//远程操作类
    NETDVR_LOGSEARCH_MAIN_EXCEPTION,//异常类
    NETDVR_LOGSEARCH_MAIN_ALL,          //所有类
} logsearch_main_t;
```

44、子类型

```
typedef enum NETDVR_LOGSEARCH_SLAVE {
    /*NETDVR_LOGSEARCH_MAIN_ALARM's slave type*/
    NETDVR_LOGSEARCH_ALARM_IN_BEGIN = 0,//报警类之报警输入开始
    NETDVR_LOGSEARCH_ALARM_IN_END,          //报警类之报警输入结束
    NETDVR_LOGSEARCH_ALARM_MD_BEGIN,    //报警类之移动侦测开始
    NETDVR_LOGSEARCH_ALARM_MD_END, //报警类之移动侦测结束
    NETDVR_LOGSEARCH_ALARM_ALL,             //报警类之所有
    /*NETDVR_LOGSEARCH_MAIN_LOCALOP's slave type*/
    NETDVR_LOGSEARCH_LOCALOP_STARTUP = 16,//本地操作之启动
    NETDVR_LOGSEARCH_LOCALOP_SHUTDOWN,      //本地操作之关闭
    NETDVR_LOGSEARCH_LOCALOP_LOGIN,         //本地操作之登录
    NETDVR_LOGSEARCH_LOCALOP_LOGOUT,            //本地操作之登出
    NETDVR_LOGSEARCH_LOCALOP_CONFIG,            //本地操作之系统配置
    NETDVR_LOGSEARCH_LOCALOP_REC_START,     //本地操作之录像开始
    NETDVR_LOGSEARCH_LOCALOP_REC_STOP,  //本地操作之录像结束
    NETDVR_LOGSEARCH_LOCALOP_UPDATE,            //本地操作之升级
    NETDVR_LOGSEARCH_LOCALOP_FORMAT,            //本地操作之格式化
    NETDVR_LOGSEARCH_LOCALOP_ALL,               //本地操作之所有
    /*NETDVR_LOGSEARCH_MAIN_REMOTEOP's slave type*/
    NETDVR_LOGSEARCH_REMOTEOP_LOGIN = 32,   //远程操作之登录
    NETDVR_LOGSEARCH_REMOTEOP_LOGOUT,  //远程本地操作之登出
    NETDVR_LOGSEARCH_REMOTEOP_REC_START,//远程操作之录像开始
    NETDVR_LOGSEARCH_REMOTEOP_VIDEO_STOP,//远程操作之录像结束
    NETDVR_LOGSEARCH_REMOTEOP_CONFIG,           //远程操作之系统配置
    NETDVR_LOGSEARCH_REMOTEOP_RESTART, //远程操作之系统重启
    NETDVR_LOGSEARCH_REMOTEOP_VOIP_START,//远程操作之语音对讲开始
    NETDVR_LOGSEARCH_REMOTEOP_VOIP_STOP,    //远程操作之语音对讲结束
    NETDVR_LOGSEARCH_REMOTEOP_UPDATE,   //远程操作之升级
    NETDVR_LOGSEARCH_REMOTEOP_ALL,              //远程操作之所有
    /*NETDVR_LOGSEARCH_MAIN_EXCEPTION's slave type*/
    NETDVR_LOGSEARCH_EXCEPTION_VLOSS = 48,  //异常之视频丢失
    NETDVR_LOGSEARCH_EXCEPTION_VBLIND,  //异常之遮挡
    NETDVR_LOGSEARCH_EXCEPTION_HDD_ERR,//异常之硬盘错误
    NETDVR_LOGSEARCH_EXCEPTION_HDD_FULL,    //异常之硬盘满
    NETDVR_LOGSEARCH_EXCEPTION_IP_CONFLICT,//异常之ip冲突
    NETDVR_LOGSEARCH_EXCEPTION_ILLEGALOP,//异常之非法操作
```

```
    NETDVR_LOGSEARCH_EXCEPTION_ALL,        //异常之所有
} logsearch_slave_t;
```

45、日志信息

```
struct NETDVR_logInfo_t
{
    unsigned int startTime;                 //创建时间
    char main[64];                          //主类型
    char slave[64];                         //子类型
    char loginfo[32];                           //日志内容
    struct NETDVR_logInfo_t *pnext;         //指针指向下一条日志
};
```

46、日志检索结果

```
struct NETDVR_logSearchResult_t
{
    unsigned short sum;                         //检索到的日志条数
    unsigned short startID;                 //返回日志从第startID条开始
    unsigned short endID;                   //到endID条结束
    struct NETDVR_logInfo_t *pH;            //指向返回的第一条日志
};
```

47、视频（图像）参数

```
struct NETDVR_videoParam_t
{
    char channelname[32];                   //通道名

    unsigned short name_x, name_y;          //通道名显示位置
    unsigned short time_x, time_y;          //通道时间显示位置
    unsigned char flag_name, name_set;      //通道名是否显示；通道名位置是否改变
    unsigned char flag_time, time_set;      //时间是否显示；显示位置是否改变
    unsigned char overlay;                  //遮挡
    unsigned char lost;                     //视频丢失（参阅六-49）
    unsigned char motion;                   //移动侦测灵敏度（参阅六-50）
    unsigned char flag_buzz;                    //是否声音警告
    unsigned char flag_send;                //reserved
    unsigned char flag_mask;                    //是否显示遮盖
    unsigned char mask_count;               //遮盖数
    struct NETDVR_MASKAREA_t **p_area;//遮盖区域数组头指针：（参阅六-48）
    char record_channel[16];                    //触发录像通道
    char alarm_out[16];                     //触发报警输出
};
```

48、遮盖区域

```
struct NETDVR_MASKAREA_t
{
    unsigned short x, y;                    //区域开始位置
    unsigned short width, height;           //遮盖宽和高
};
```

49、视频丢失是否处理

```
#define NETDVR_VIDEO_LOST_DECT 0 //处理
#define NETDVR_VIDEO_LOST_UNDECT 1   //不处理
```

50、移动侦测处理方式

```
#define NETDVR_VIDEO_MD_CLOSE 0                 //关闭移动侦测
#define NETDVR_VIDEO_MD_SEN_LOWEST 1            //最低灵敏度处理
#define NETDVR_VIDEO_MD_SEN_LOWER 2             //更低灵敏度处理
#define NETDVR_VIDEO_MD_SEN_LOW 3               //低灵敏度处理
#define NETDVR_VIDEO_MD_SEN_HIGH 4              //高灵敏度处理
#define NETDVR_VIDEO_MD_SEN_HIGHER 5            //更高灵敏度处理
#define NETDVR_VIDEO_MD_SEN_HIGHEST 6           //最高灵敏度处理
```

51、录像参数

```
struct NETDVR_recordParam_t
{
    char channelname[32];                       //不可修改
    flow_type_t code_type;                      //码流类型（参阅六-52）
    bitrate_type_t bit_type;                    //位率类型（参阅六-53）
    bitrate_t bit_max;                          //位率（参阅六-54）
    video_quality_t quality;                    //图象质量（参阅六-55）
    framerate_t frame_rate;                     //视频帧率（参阅六-56）
    prerecord_time_t pre_record;                //预录时间（参阅六-57）
    postrecord_time_t post_record;              //录像延时时间（参阅六-58）
    video_format_t yfactor;                     //视频制式（参阅六-26）
    vga_resolution_t video_resolution;          //vga分辨率（参阅六-27）
    unsigned char flag_record;                  //是否启用录像
    subctrl_t sub_ctrl;                         //子码流控制方式（参阅六-59）
    framerate_t sub_framerate;                  //子码流帧率（参阅六-56）
    bitrate_t sub_bitrate;                      //子码流位率（参阅六-54）
};
```

52、媒体流类型

```
typedef enum NETDVR_FLOW_TYPE {
    NETDVR_FLOW_VIDEO = 0,                      //视频流
    NETDVR_FLOW_MUTI,                           //复合流
} flow_type_t;
```

53、位率类型

```
typedef enum NETDVR_BITRATE_TYPE {
    NETDVR_BITRATE_FIXED = 0,          //定码率
    NETDVR_BITRATE_VARIABLE,           //变码率
} bitrate_type_t;
```

54、位率

```
typedef enum NETDVR_BITRATE {
    NETDVR_BITRATE_64 = 64,            //64kbps
    NETDVR_BITRATE_128 = 128,          //128kbps
    NETDVR_BITRATE_256 = 256,          //256kbps
    NETDVR_BITRATE_384 = 386,          //386kbps
    NETDVR_BITRATE_512 = 512,          //512kbps
    NETDVR_BITRATE_768 = 768,          //768kbps
    NETDVR_BITRATE_1024 = 1024,        //1Mbps
    NETDVR_BITRATE_1536 = 1536,        //1.5Mbps
    NETDVR_BITRATE_2048 = 2048,        //2Mbps
} bitrate_t;
```

55、图象质量（在位率类型为NETDVR_BITRATE_VARIABLE下有效）

```
typedef enum NETDVR_VIDEO_QUALITY {
    NETDVR_VIDEO_QUALITY_BEST = 0,     //最好
    NETDVR_VIDEO_QUALITY_BETTER,       //更好
    NETDVR_VIDEO_QUALITY_GOOD,         //好
    NETDVR_VIDEO_QUALITY_NORMAL,       //一般
    NETDVR_VIDEO_QUALITY_BAD,          //差
    NETDVR_VIDEO_QUALITY_WORSE,        //更差
} video_quality_t;
```

56、视频帧率

```
typedef enum NETDVR_FRAMERATE {
    NETDVR_FRAMERATE_25 = 25,          //25f/s
    NETDVR_FRAMERATE_20 = 20,          //20f/s
    NETDVR_FRAMERATE_15 = 15,          //15f/s
    NETDVR_FRAMERATE_10 = 10,          //10f/s
    NETDVR_FRAMERATE_5 = 5,            //5f/s
    NETDVR_FRAMERATE_2 = 2,            //2f/s
    NETDVR_FRAMERATE_1 = 1,            //1f/s
} framerate_t;
```

57、预录时间

```
typedef enum NETDVR_PRERECORD_TIME {
    NETDVR_PRERECORD_TIME_0 = 0,       //不预录
    NETDVR_PRERECORD_TIME_5,           //5s
```

```
    NETDVR_PRERECORD_TIME_10,          //10s
    NETDVR_PRERECORD_TIME_15,          //15s
    NETDVR_PRERECORD_TIME_20,          //20s
    NETDVR_PRERECORD_TIME_25,          //25s
    NETDVR_PRERECORD_TIME_30,          //30s
    NETDVR_PRERECORD_TIME_60,          //60s
} prerecord_time_t;
```

58、录像延时时间

```
typedef enum NETDVR_POSTRECORD_TIME {
    NETDVR_POSTRECORD_TIME_0 = 0,      //不延时
    NETDVR_POSTRECORD_TIME_5,          //5s
    NETDVR_POSTRECORD_TIME_10,         //10s
    NETDVR_POSTRECORD_TIME_30,         //30s
    NETDVR_POSTRECORD_TIME_60,         //60s
    NETDVR_POSTRECORD_TIME_120,        //120s
    NETDVR_POSTRECORD_TIME_300,        //300s
    NETDVR_POSTRECORD_TIME_600,        //600s
} postrecord_time_t;
```

59、子码流控制方式

```
typedef enum NETDVR_SUBCTRL_TYPE {
    NETDVR_SUBCTRL_AUTO = 0,           //自动控制
    NETDVR_SUBCTRL_MANUAL,             //手动
} subctrl_t;
```

60、录像布防参数

```
struct NETDVR_recordTimeParam_t
{
    timetype_t flag_alltime;           //布防时间类型（参阅六-61）
    recordtype_t alltype;              //全天布防录像类型（参阅六-63）
    unsigned int start_time[4];        //分段布防开始时间数组
    unsigned int end_time[4];          //分段布防结束时间数组
    recordtype_t type[4];              //分段布防结束类型数组（参阅六-63）
};
```

61、布防时间类型

```
typedef enum NETDVR_TIMETYPE {
    NETDVR_TIMETYPE_WHOLEDAY = 0,      //全天布防
    NETDVR_TIMETYPE_NONE,              //全天不布防
    NETDVR_TIMETYPE_SEGMENT,           //分时段布防
} timetype_t;
```

62、星期定义

```
typedef enum NETDVR_WEEKDAY {
    NETDVR_WEEKDAY_1 = 0,                    //星期一
    NETDVR_WEEKDAY_2,                        //星期二
    NETDVR_WEEKDAY_3,                        //星期三
    NETDVR_WEEKDAY_4,                        //星期四
    NETDVR_WEEKDAY_5,                        //星期五
    NETDVR_WEEKDAY_6,                        //星期六
    NETDVR_WEEKDAY_7,                        //星期日
} weekday_t;
```

63、录像类型

```
typedef enum NETDVR_RECORDTYPE {
    NETDVR_RECORDTYPE_SCHEDULE = 0,          //定时
    NETDVR_RECORDTYPE_MD,                    //移动侦测
    NETDVR_RECORDTYPE_ALARM,                 //报警
} recordtype_t;
```

64、异常类型

```
typedef enum NETDVR_EXCEPTIONTYPE {
    NETDVR_EXCEPTION_VLOSS = 0,              //视频丢失
    NETDVR_EXCEPTION_VBLIND,                 //遮挡
    NETDVR_EXCEPTION_HDD_FULL,               //硬盘满l
    NETDVR_EXCEPTION_HDD_ERR,                //硬盘出错
    NETDVR_EXCEPTION_LINK_LOST,              //网络断开
    NETDVR_EXCEPTION_IP_CONFLICT,            //IP冲突
} exceptiontype_t;
```

65、异常处理参数

```
struct NETDVR_exceptionParam_t
{
    unsigned char flag_display;              //保留
    unsigned char flag_buzz;                     //是否声音警告.
    unsigned char flag_send;                 //保留
    unsigned char flag_alarmout;                 //是否触发报警输出
    unsigned char alarm_out[4];              //保留
};
```

66、报警输入参数

```
struct NETDVR_alarmInParam_t
{
    unsigned char flag_deal;                 //是否处理报警输入
    unsigned char in_copy;                   //是否复制报警输入参数
    unsigned char copyin;                    //复制参数到copyin路报警输入
    alarmintype_t typein;                        //报警器类型（参阅六-67）
```

```
};
```

67、报警器类型

```
typedef enum NETDVR_ALARMINTYPE {
    NETDVR_ALARMIN_HIGH = 1,            //高电平报警
    NETDVR_ALARMIN_LOW,                 //低电平报警
} alarmintype_t;
```

68、布防参数

```
struct NETDVR_alarmSetTime_t
{
    timetype_t flag_alltime;            //布防时间类型（参阅六-61）
    unsigned int starttime[4];          //分段布防开始时间数组
    unsigned int endtime[4];            //分段布防结束时间数组
};
```

69、报警处理参数

```
struct NETDVR_alarmHandler_t
{


        unsigned char flag_buzz;            //是否声音报警
        unsigned char flag_send;            //保留
        unsigned char flag_alarmout;            //触发报警输出
        unsigned char alarm_out[4];         //报警输出设置
        unsigned char reserved[28];
        unsigned char recchan[NETDVR_MAX_CHN_R6104];            //触发报警通道
        unsigned char reserved2[28];


};
```

70、PTZ联动参数

```
struct NETDVR_alarmInPtz_t
{
    unsigned char channo;               //PTZ所在通道
    unsigned char flag_preset;          //是否启用预制点
    unsigned char preset;               //预制点号
    unsigned char flag_cruise;          //是否启用巡航
    unsigned char cruise;               //巡航号
    unsigned char flag_track;           //是否启用轨迹
};
```

71、报警输出参数

```
struct NETDVR_alarmOutParam_t
```

```
{
    unsigned char out_copy;                 //是否复制报警输出参数
    unsigned char copyout;                  //复制参数到copyout路报警输入
    alarmouttype_t typeout;                 //报警器类型（参阅六-72）
    alarmoutdelay_t delay;                  //报警输出延时（参阅六-73）
};
```

72、输出报警器类型

```
typedef enum NETDVR_ALARMOUTYPE {
    NETDVR_ALARMOUT_NO = 1,             //常开
    NETDVR_ALARMOUT_NC,                 //常闭
} alarmouttype_t;
```

73、报警输出延时

```
typedef enum NETDVR_ALARMOUTDELAY {
    NETDVR_ALARMOUTDELAY_5 = 0,         //5s
    NETDVR_ALARMOUTDELAY_10,            //10s
    NETDVR_ALARMOUTDELAY_30,            //30s
    NETDVR_ALARMOUTDELAY_60,            //60s
    NETDVR_ALARMOUTDELAY_120,       //120s
    NETDVR_ALARMOUTDELAY_300,       //300s
    NETDVR_ALARMOUTDELAY_600,       //600s
    NETDVR_ALARMOUTDELAY_MANUAL,    //manual
} alarmoutdelay_t;
```

74、手动录像参数

```
struct NETDVR_manualRecordParam_t
{
        struct
        {
            unsigned char allrec;                                   //全部启动
            unsigned char allstop;                                  //全部停止
            unsigned char rec_status[NETDVR_MAX_RECORD_CHN]; //单独每路的
        }ManualRecordParam;

    unsigned char reserved[16];   //保留字段

    };
```

75、网络参数

```
struct NETDVR_networkParam_t
{
```

```
    char mac_address[18];                   //mac地址
    unsigned int ip_address;                //ip地址
    unsigned short server_port;             //服务器控制端口
    unsigned int net_mask;                  //子网掩码
    unsigned int net_gateway;                   //网关ip
    unsigned int dns;                           //dns服务器的ip地址
    unsigned int multicast_address;         //组播地址
    unsigned int admin_host;                    //中心服务器
    unsigned short host_port;                   //主机端口号
    unsigned short http_port;                   //http服务端口
    unsigned char flag_pppoe;               //是否启用pppoe
    char pppoe_user_name[64];                       //pppoe帐号
    char pppoe_passwd[64];                          //pppoe密码
    char pppoe_pwdcheck[64];                        //pppoe确认密码
    unsigned char flag_dhcp;                    //是否启用dhcp
    unsigned char flag_ddns;                    //是否启用ddns
    char ddns_domain[64];                   //ddns服务器
    char ddns_user[64];                     //ddns帐号
    char ddns_passwd[64];                   //ddns密码
};
```

76、升级参数

```
struct NETDVR_updateParam_t
{
    char filename[64];                      //文件名（非必要）
    unsigned int size;                          //升级文件大小
    unsigned int verify;                    //校验码
    unsigned short version;                 //升级版本
    unsigned short reserved;                //升级类型（参阅六-77）
NETDVR_UPDATE_MOTHERBBOARD/NETDVR_UPDATE_PANEL
};
```

77、升级类型

```
#define NETDVR_UPDATE_MOTHERBBOARD 0//主板程序（内核）升级
#define NETDVR_UPDATE_PANEL 1               //控制面板升级
```

78、移动侦测参数

```
struct NETDVR_motionDetection_t
{
    union
    {
        struct
        {
            unsigned char flag;                 //是否打开移动侦测
```

```
            unsigned char sense;                    //灵敏度值
            unsigned char block[NETDVR_MD_ROW_R6104*NETDVR_MD_COL_R6104];//区域选择值
        }R6104;

        struct
        {
            unsigned char flag;                     //
            unsigned char sense;                    //
            unsigned char block[NETDVR_MD_ROW_R8016*NETDVR_MD_COL_R8016];//
        }R8016;

        {
            unsigned char flag;
            unsigned char sense;
            unsigned char block[NETDVR_MD_ROW_R9000*NETDVR_MD_COL_R9000];
        } R9000;
    };
};
```

79、移动侦测宏

```
#define NETDVR_MD_ROW_R6104    18                   //移动侦测区域划分出的行数
#define NETDVR_MD_COL_R6104 22                      //移动侦测区域划分出的列数
#define NETDVR_MD_ROW_R8016        12
#define NETDVR_MD_COL_R8016        16
#define NETDVR_MD_ROW_R9000        18
#define NETDVR_MD_COL_R9000        22
#define NETDVR_MD_MIN_SENSE  0              //移动侦测最小灵敏度
#define NETDVR_MD_MAX_SENSE 5               //移动侦测最大灵敏度
```

80、用户列表头

```
struct NETDVR_userList_t
{
    unsigned short     sum;                 //用户总数
    unsigned short     startID;                 //从第startID个用户开始返回
    unsigned short     endID;                   //到第endID个用户结束
    struct NETDVR_userInfo_t *p_head;       //指针指向返回的用户列表（参阅81）
};
```

81、用户资料

```
struct NETDVR_userInfo_t
{
    char name[12];                              //用户名
    char   password[12];                        //密码
    char confirm[12];                           //密码确认
```

```
    char mac_address[18];                   //mac地址

    /* 1:权限打开， 0:权限关闭 */
    unsigned int rcamer;                        //远程云台控制
    unsigned int rrec;                          //远程录像控制
    unsigned int rplay;                     //远程回放
    unsigned int rsetpara;                      //远程参数设置
    unsigned int rlog;                      //远程日志访问
    unsigned int rtool;                     //远程管理工具
    unsigned int rpreview;                  //远程预览
    unsigned int ralarm;                        //远程报警管理
    unsigned int rvoip;                     //远程语音对讲
    unsigned int lcamer;                        //本地云台控制
    unsigned int lrec;                          //本地录像控制
    unsigned int lplay;                     //本地回放
    unsigned int lsetpara;                      //本地参数设置
    unsigned int llog;                      //本地日志访问
    unsigned int ltool;                     //本地管理工具
    struct NETDVR_userInfo_t *pnext;        //指针指向下一个用户
};
```

82. 远程硬盘信息

```
struct NETDVR_hddInfo_t //结构体
{
    struct    NETDVR_eachHdd_t //每个硬盘的信息
    {
        unsigned char  hdd_exist;    //1 exist; 0 not exist  硬盘是否存在
        __int64        capability;          //MB  硬盘总容量，单位MB
        __int64        freesize;       //MB  硬盘剩余容量  单位  MB
    } each_hddInfo[8];
};
```

83. 系统版本信息

```
struct NET_SysVerInfo_t
{
    char devicename[32]; //设备名
    char devicemodel[32]; //设备型号
    char deviceser[32]; //设备序列号
    char version[64]; //设备版本号
};
```

84.报警状态

```
//报警状态枚举
typedef enum NETDVR_ALARM_STATE
```

```
{
    NETDVR_ALARMSTATE_CLEAR,    //无报警
    NETDVR_ALARMSTATE_SET       //有报警
}alarmState_t;

//报警输入输出状态
//目前输出状态只支持一路报警输出
struct NETDVR_AlarmIOState_t
{
    alarmState_t alarmInState[4];
    alarmState_t alarmOutState[4];          //only support alarmOutState[0] now
};

85子码流参数
struct NETDVR_subVideoParam_t
{
    unsigned int ip;     .//.要访问的ip
    unsigned char chn; //要访问的通道
};
```

七、 函数定义

函数返回值含义，请查阅 NETDVR_RETURN_CODE。

1、 int NETDVR_startup(void);

功能	初始化客户端 SDK 动态库
参数	无
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	在初始化 DVR 池时使用一次。

2、 int NETDVR_cleanup(void);

功能	清理和释放 DVR 池使用的资源
参数	无
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	在程序退出时使用一次。

3、 int NETDVR_createDVR(int *p_handle,struct NETDVR_serverInfo_t * p_para);

功能	创建一个 DVR 客户端并建立与 DVR 服务器的连接
参数	p_handle：用于返回 DVR HANDLE 的指针（p_handle 不能为空） p_para：DVR 客户端连接的服务器信息（参阅六-1）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	在七-1 后使用

4、int NETDVR_destroyDVR(int Handle);

功能	销毁一个 DVR 客户端
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-3 后可以使用

5、int NETDVR_regCBMsgConnLost(int Handle, PFUN_MSG_T p_cb_func, unsigned int dwContent);

功能	注册 DVR 客户端与服务器断连回调函数
参数	Handle：七-3 返回的 DVR HANDLE p_cb_func：消息回调函数（参阅六-2） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	紧接着七-3 后调用

6、int NETDVR_loginServer(int Handle, const struct NETDVR_loginInfo_t * p_para);

功能	登录服务器
参数	Handle：七-3 返回的 DVR HANDLE p_para：登录信息（参阅六-3）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-3 后可以使用

7、int NETDVR_logoutServer(int Handle);

功能	登出服务器
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-6 使用有效

8、int NETDVR_loginEditPass(int Handle, const struct NETDVR_loginPass_t *p_para);

功能	密码修改
参数	Handle：七-3 返回的 DVR HANDLE p_para：密码修改信息（参阅六-4）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-6 使用有效

9、int NETDVR_openVideoReciever(int Handle, const struct NETDVR_mediaRcvParam_t *p_rcv_para, pFrameCallBack pCBFun, unsigned int dwContent);

功能	打开视频通道主码流接收

参数	Handle：七-3 返回的 DVR HANDLE p_rcv_para：码流接收参数（参阅六-5） pCBFun,：原始码流接收回调函数（参阅六-6） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

10、int NETDVR_closeVideoReciever(int Handle, unsigned char rcv_chn);

功能	关闭视频通道主码流接收
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：接收通道号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-9 使用有效

11、int NETDVR_setDecoderFMT(int Handle, unsigned char rcv_chn, fmt_type_t fmt);

功能	设置视频通道主码流接收解码格式
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：通道号 fmt：解码格式（参阅六-8）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

12、int NETDVR_createVideoDecoder(int Handle, unsigned char rcv_chn, pDecFrameCallBack pCBFun, unsigned int dwContent);

功能	创建视频通道主码流解码器
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：通道号 pCBFun：解码后数据回调函数（参阅六-9） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

13、int NETDVR_destroyVideoDecoder(int Handle, unsigned char rcv_chn);

功能	销毁视频通道主码流解码器
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：通道号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

14、int NETDVR_startVideoSend(int Handle, const struct NETDVR_mediaRcvParam_t *p_rcv_para);

功能	请求主机端开始发送视频主码流
参数	Handle：七-3 返回的 DVR HANDLE p_rcv_para：主码流发往七-9 设定的 ip 和端口（参阅六-5）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-9 执行成功后使用最佳（否则会丢帧）

15、int NETDVR_stopVideoSend(int Handle, unsigned char rcv_chn);

功能	请求主机端停止发送视频主码流
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：通道号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

16、int NETDVR_snapshot(int Handle, int chn, char *path, char *filename);

功能	抓图
参数	Handle：七-3 返回的 DVR HANDLE chn：通道号 path：保存目录 filename：保存文件名
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-9，12，14 成功执行后

17、int NETDVR_openAudioReciever(int Handle, const struct NETDVR_mediaRcvParam_t *p_rcv_para, pFrameCallBack pCBFun, unsigned int dwContent);

功能	打开音频通道码流接收
参数	Handle：七-3 返回的 DVR HANDLE p_rcv_para：码流接收参数（参阅六-5） pCBFun,：原始码流接收回调函数（参阅六-6） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

18、int NETDVR_closeAudioReciever(int Handle, unsigned char rcv_chn);

功能	关闭音频通道码流接收
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：接收通道号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-17 使用有效

19、int NETDVR_startAudioSend(int Handle, const struct NETDVR_mediaRcvParam_t *p_rcv_para);

功能	请求主机端开始发送音频流
参数	Handle：七-3 返回的 DVR HANDLE p_rcv_para：音频流发往七-18 设定的 ip 和端口（参阅六-5）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-17 执行成功后使用最佳（否则会丢帧）

20、int NETDVR_stopAudioSend(int Handle, unsigned char rcv_chn);

功能	请求主机端停止发送音频流
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：通道号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

21、int NETDVR_recFilesSearch(int Handle, const struct NETDVR_fileSearchCondition_t *prfs, struct NETDVR_recFileSearchResult_t *pdesc);

功能	DVR 已录文件检索
参数	Handle：七-3 返回的 DVR HANDLE prfs：检索条件（参阅六-11） pdesc：检索结果（参阅六-13）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

22、int NETDVR_recFilesSearchClean(struct NETDVR_recFileSearchResult_t *pdesc);

功能	释放 DVR 已录文件检索结果所用资源
参数	pdesc：检索结果（参阅六-13）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-21 成功执行后

23、int NETDVR_regCBFileRecieverProgress(int Handle, PFUN_PROGRESS_T p_cb_progress, unsigned int dwContent);

功能	注册文件下载接收进度回调函数
参数	Handle：七-3 返回的 DVR HANDLE p_cb_progress：下载录像文件接收进度回调函数（参阅六-15） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

24、int NETDVR_regCBFileRecieverError(int Handle, PFUN_ERROR_T p_cb_err, unsigned int dwContent);

功能	注册文件下载接收线程错误回调函数
参数	Handle：七-3 返回的 DVR HANDLE p_cb_err：下载录像文件接收线程错误回调函数（参阅六-17） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

25、int NETDVR_setFileReciever(int Handle, char *s_save_dir, char *s_save_filename, unsigned int rcv_filesize);

功能	设置文件下载接收参数
参数	Handle：七-3 返回的 DVR HANDLE s_save_dir：下载后文件保存目录 s_save_filename：下载后文件保存名称 rcv_filesize：下载文件大小
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

26、int NETDVR_openFileReciever(int Handle, unsigned int rcv_ip, unsigned short rcv_port);

功能	打开文件下载接收线程
参数	Handle：七-3 返回的 DVR HANDLE rcv_ip：接收 ip rcv_port：接收端口
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

27、int NETDVR_closeFileReciever(int Handle);

功能	关闭文件下载接收线程
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-21 执行成功后

28、int NETDVR_startFileDownload(int Handle, unsigned int rcv_ip, unsigned short rcv_port, const struct NETDVR_recFileInfo_t *pFileInfo);

功能	请求主机端开始发送需要下载的文件
参数	Handle：七-3 返回的 DVR HANDLE rcv_ip：接收文件下载数据的 ip rcv_port：接收文件下载数据的端口 pFileInfo：请求下载的文件（参阅六-14）

返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	七-26 执行成功后

29、int NETDVR_stopFileDownload(int Handle);

功能	请求主机端停止发送需要下载的文件
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

30、int NETDVR_openPlayCBReciever(int Handle, const struct NETDVR_playerRcvPara_t *p_player_info, pDecFrameCallBack pPlayerCBFun, unsigned int dwContent);

功能	打开回放码流接收
参数	Handle：七-3 返回的 DVR HANDLE p_player_info：回放码流接收参数（参阅六-18） pPlayerCBFun,：回放码流接收解码后回调函数（参阅六-9） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

31、int NETDVR_setPlayCBDecoderFMT(int Handle, int player_index, fmt_type_t fmt);

功能	设置回放视频流接收解码格式
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 fmt：解码格式（参阅六-8）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

32、int NETDVR_closePlayCBReciever(int Handle, int player_index);

功能	关闭回放码流接收
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

33、int NETDVR_regCBMsgPlayOver(int Handle, int player_index, PFUN_MSG_T p_cb_func, unsigned int dwContent);

功能	注册回放结束消息回调函数
参数	Handle：七-3 返回的 DVR HANDLE

	player_index：回放设备号 p_cb_func：回放结束消息回调函数（参阅六-19） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

34、int NETDVR_regCBMsgHasAudio(int Handle, int player_index, PFUN_MSGHASAUDIO_T p_cb_func, unsigned int dwContent);

功能	注册回放文件是否含音频消息回调函数
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 p_cb_func：回放是否含音频消息回调函数（参阅六-20） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

35、int NETDVR_regCBMsgProgress(int Handle, int player_index, PFUN_PROGRESS_T p_cb_func, unsigned int dwContent);

功能	注册回放文件是否含音频消息回调函数
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 p_cb_func：回放进度消息回调函数（参阅六-15） dwContent：用户自定义回调函数参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

36、int NETDVR_startFilePlay(int Handle, const struct NETDVR_playerRcvPara_t *p_player_info, const struct NETDVR_recFileInfo_t *pFileInfo);

功能	请求服务器开始发送指定的回放文件
参数	Handle：七-3 返回的 DVR HANDLE p_player_info：回放码流接收参数（参阅六-18） pFileInfo：回放文件（参阅六-14）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

37、int NETDVR_stopFilePlay(int Handle, int player_index);

功能	请求服务器结束发送回放文件
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。

前提	略

38、int NETDVR_startTimePlay(int Handle, const struct NETDVR_playerRcvPara_t *p_player_info, const struct NETDVR_fileSearchCondition_t *prfs);

功能	请求服务器开始按时间和通道发送回放文件（按时间回放）
参数	Handle：七-3 返回的 DVR HANDLE p_player_info：回放码流接收参数（参阅六-18） prfs：回放搜索条件（参阅六-11）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

39、int NETDVR_stopTimePlay(int Handle, int player_index);

功能	请求服务器停止时间回放
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

40、int NETDVR_pausePlay(int Handle, int player_index);

功能	请求服务器暂停回放码流发送
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

41、int NETDVR_resumePlay(int Handle, int player_index);

功能	请求服务器继续回放码流发送
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

42、int NETDVR_singleFramePlay(int Handle, int player_index);

功能	请求服务器按单帧回放码流发送
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

43、int NETDVR_fastPlay(int Handle, int player_index);

功能	向服务器发送快放请求
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

44、int NETDVR_slowPlay(int Handle, int player_index);

功能	向服务器发送慢放请求
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

45、int NETDVR_setPlayRate(int Handle, int player_index, int play_rate);

功能	控制服务器回放发送速率
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 play_rate：回放速率（[-8, 8]）-n 表示 1/n 正常速率
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

46、int NETDVR_playPrevious(int Handle, int player_index);

功能	请求服务器回放上一段（按文件）或者上一文件（按时间）
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

47、int NETDVR_playNext(int Handle, int player_index);

功能	请求服务器回放下一段（按文件）或者下一文件（按时间）
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

48、int NETDVR_playSeek(int Handle, int player_index, unsigned int new_time);

功能	请求服务器回放从指定位置开始回放
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 new_time：0 到回放总时间之间
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

49、int NETDVR_playMute(int Handle, int player_index, int b_mute);

功能	请求服务器静音或者非静音回放
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 b_mute：1-静音；0-非静音
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

50、int NETDVR_playProgress(int Handle, int player_index, int b_send_progress);

功能	控制服务发送或者不发送回放进度
参数	Handle：七-3 返回的 DVR HANDLE player_index：回放设备号 b_send_progress：1-发送；0-不发送
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

51、int NETDVR_setSystemParams(int Handle, const struct NETDVR_systemParam_t *pSysPara);

功能	设置系统参数
参数	Handle：七-3 返回的 DVR HANDLE pSysPara：新系统参数（参阅六-21）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

52、int NETDVR_getSystemParams(int Handle, struct NETDVR_systemParam_t *pSysPara);

功能	获得系统参数
参数	Handle：七-3 返回的 DVR HANDLE pSysPara：用于返回已有系统参数（参阅六-21）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

53、int NETDVR_setPtzParams(int Handle, unsigned char chn, const struct NETDVR_ptzParam_t *ptzParam);

功能	获得云台参数
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 ptzParam：用于返回已有云台参数（参阅六-30）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

54、int NETDVR_getPtzParams(int Handle, unsigned char chn, struct NETDVR_ptzParam_t *p_ptz_param);

功能	设置云台参数
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 p_ptz_param：新的云台参数（参阅六-30）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

55、int NETDVR_startPtzControl(int Handle, unsigned char chn, enum NETDVR_PTZCONTROL ptz_cmd);

功能	开始控制云台
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 ptz_cmd：控制命令（参阅六-37）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

56、int NETDVR_stopPtzControl(int Handle, unsigned char chn);

功能	停止控制云台
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

57、int NETDVR_setYuntaiPresetPoint(int Handle, unsigned char chn, unsigned char preset_pos);

功能	设置云台预制点
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 preset_pos：预制点编号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

58、int NETDVR_goYuntaiPresetPoint(int Handle, unsigned char chn, unsigned char preset_pos);

功能	转到云台预制点
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 preset_pos：预制点编号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

59、int NETDVR_clearYuntaiPresetPoint(int Handle, unsigned char chn, unsigned char preset_pos);

功能	清除云台预制点
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 preset_pos：预制点编号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

60、int NETDVR_startRecordYuntaiTrack(int Handle, unsigned char chn);

功能	启动通道云台轨迹记录
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

61、int NETDVR_stopRecordYuntaiTrack(int Handle, unsigned char chn);

功能	停止通道云台轨迹记录
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

62、int NETDVR_startYuntaiTrack(int Handle, unsigned char chn);

功能	开始运行通道云台轨迹
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

63、int NETDVR_stopYuntaiTrack(int Handle, unsigned char chn);

功能	停止运行通道云台轨迹

参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

64、int NETDVR_insertYuntaiCruisePos(int Handle, unsigned char chn, struct NETDVR_cruisePath_t *p_cruise_path, const struct NETDVR_cruisePos_t *p_cruise_pos);

功能	为巡航路径添加巡航点
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 p_cruise_path：巡航路径（参阅六-39） p_cruise_pos：巡航点（参阅六-40）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

65、int NETDVR_deleteYuntaiCruisePos(int Handle, unsigned char chn, struct NETDVR_cruisePath_t *p_cruise_path, unsigned char cruise_index);

功能	为巡航路径删除巡航点
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 p_cruise_path：巡航路径（参阅六-39） cruise_index：巡航点号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

66、 int NETDVR_startYuntaiCruise(int Handle, unsigned char chn, unsigned char path_no);

功能	开始路径巡航
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 path_no：巡航路径号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

I 67、nt NETDVR_stopYuntaiCruise(int Handle, unsigned char chn, unsigned char path_no);

功能	停止路径巡航
参数	Handle：七-3 返回的 DVR HANDLE chn：云台视频输入通道 path_no：巡航路径号
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。

前提	略

68、int NETDVR_startVOIP(int Handle, const struct NETDVR_mediaRcvParam_t *p_rcv_para);

功能	开始语音对讲
参数	Handle：七-3 返回的 DVR HANDLE p_rcv_para：语音对讲接收参数（参阅六-5）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

69、int NETDVR_stopVOIP(int Handle, unsigned char rcv_chn);

功能	停止语音对讲
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：语音对通道号（只有一个用 0）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

70、int NETDVR_startRecord(int Handle, unsigned char chn, char *p_dir_path, unsigned int file_max_len);

功能	开始客户端录像
参数	Handle：七-3 返回的 DVR HANDLE chn：录像通道 p_dir_path：录像文件保存目录 file_max_len：录像文件大小限制
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

71、int NETDVR_stopRecord(int Handle, unsigned char chn);

功能	停止客户端录像
参数	Handle：七-3 返回的 DVR HANDLE chn：录像通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

72、int NETDVR_setRecordCB(int Handle, unsigned char chn, pFrameCallBack pRecordCBFun, unsigned int dwContent);

功能	客户端录像回调函数
参数	Handle：七-3 返回的 DVR HANDLE chn：录像通道 pRecordCBFun：用户自定义录像处理回调函数（参阅六-6）

	dwContent：用户自定义回调参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	若客户需要自行处理录像，则在七-71 前执行该函数

73 、 int NETDVR_setRecordFileNameCB(int Handle, unsigned char chn, pRecFilenameCallBack pRecFilenameCBFunc, unsigned int dwContent);

功能	客户端录像文件名回调函数
参数	Handle：七-3 返回的 DVR HANDLE chn：录像通道 pRecFilenameCBFunc：用户自定义录像处理回调函数（参阅六-6） dwContent：用户自定义回调参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	在 SDK 自动处理录像的情况下，若客户需要自定义文件名，则在七-71 前执行该函数

74 、 int NETDVR_logSearch(int Handle, struct NETDVR_logSearchCondition_t *p_condition, struct NETDVR_logSearchResult_t *p_result);

功能	Log 检索
参数	Handle：七-3 返回的 DVR HANDLE p_condition：检索条件（参阅六-41） p_result：检索结果（参阅六-46）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

75、int NETDVR_logSearchClean(struct NETDVR_logSearchResult_t *p_result);

功能	清除 Log 检索
参数	p_result：检索结果（参阅六-46）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

76、int NETDVR_setVideoParams(int Handle, unsigned char chn, const struct NETDVR_videoParam_t *pvp);

功能	设置视频参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 pvp：新的视频参数（参阅六-47）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

77、int NETDVR_getVideoParams(int Handle, unsigned char chn, struct NETDVR_videoParam_t *pvp);

功能	获取视频参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 pvp：接收服务器返回的视频参数（参阅六-47）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

78、int NETDVR_setRecordParams(int Handle, unsigned char chn, const struct NETDVR_recordParam_t *p_para);

功能	设置录像参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：新的录像参数（参阅六-51）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

79、int NETDVR_getRecordParams(int Handle, unsigned char chn, struct NETDVR_recordParam_t *p_para);

功能	获取录像参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：接收服务器返回的录像参数（参阅六-51）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

80、int NETDVR_setRecordTimeParams(int Handle, unsigned char chn, weekday_t weekday, const struct NETDVR_recordTimeParam_t *p_para);

功能	设置录像布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 weekday：星期几（参阅六-62） p_para：新的录像布防参数（参阅六-60）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

81、int NETDVR_getRecordTimeParams(int Handle, unsigned char chn, weekday_t weekday, struct NETDVR_recordTimeParam_t *p_para);

功能	获取录像布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 weekday：星期几（参阅六-62） p_para：接收服务器返回的录像布防参数（参阅六-60）

返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

82 、 int NETDVR_setExceptionParams(int Handle, exceptiontype_t except_type, const struct NETDVR_exceptionParam_t *p_para);

功能	设置异常处理参数
参数	Handle：七-3 返回的 DVR HANDLE except_type：异常类型参阅六-64） p_para：新的异常处理参数（参阅六-65）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

83、int NETDVR_getExceptionParams(int Handle, exceptiontype_t except_type, struct NETDVR_exceptionParam_t *p_para);

功能	获取异常处理参数
参数	Handle：七-3 返回的 DVR HANDLE except_type：异常类型参阅六-64） p_para：接收服务器返回的异常处理参数（参阅六-65）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

84、int NETDVR_setAlarmInParams(int Handle, unsigned char in_id, const struct NETDVR_alarmInParam_t *p_para);

功能	设置报警输入参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：新的报警输入参数（参阅六-66）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

85、int NETDVR_getAlarmInParams(int Handle, unsigned char in_id, struct NETDVR_alarmInParam_t *p_para);

功能	获取报警输入参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：接收服务器返回的报警输入参数（参阅六-66）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

86 、 int NETDVR_getAlarmInTime(int Handle, unsigned char in_id, weekday_t weekday, struct NETDVR_alarmSetTime_t *p_para);

功能	获取报警输入布防参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 weekday：星期几（参阅六-62） p_para：接收服务器返回的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

87 、 int NETDVR_setAlarmInTime(int Handle, unsigned char in_id, weekday_t weekday, const struct NETDVR_alarmSetTime_t *p_para);

功能	设置报警输入布防参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 weekday：星期几（参阅六-62） p_para：新的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

88、int NETDVR_getAlarmInAlarm(int Handle, unsigned char in_id, struct NETDVR_alarmHandler_t *p_para);

功能	获取报警输入报警处理参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：接收服务器返回的报警处理参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

89、int NETDVR_setAlarmInAlarm(int Handle, unsigned char in_id, const struct NETDVR_alarmHandler_t *p_para);

功能	设置报警输入报警处理参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：新的报警处理参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

90、int NETDVR_getAlarmInPtz(int Handle, unsigned char in_id, struct NETDVR_alarmInPtz_t *p_para);

功能	获取报警输入 PTZ 联动参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：接收服务器返回的 PTZ 联动参数（参阅六-70）

返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

91、int NETDVR_setAlarmInPtz(int Handle, unsigned char in_id, const struct NETDVR_alarmInPtz_t *p_para);

功能	设置报警输入 PTZ 联动参数
参数	Handle：七-3 返回的 DVR HANDLE in_id：第 in_id 路报警输入 p_para：新的报警 PTZ 联动参数（参阅六-70）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

92、int NETDVR_setAlarmOutParams(int Handle, unsigned char out_id, const struct NETDVR_alarmOutParam_t *p_para);

功能	设置报警输出参数
参数	Handle：七-3 返回的 DVR HANDLE out_id：第 out_id 路报警输出 p_para：新的报警输出参数（参阅六-71）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

93、int NETDVR_getAlarmOutParams(int Handle, unsigned char out_id, struct NETDVR_alarmOutParam_t *p_para);
int NETDVR_clearAlarms(int Handle);

功能	获取报警输出参数
参数	Handle：七-3 返回的 DVR HANDLE out_id：第 out_id 路报警输出 p_para：接收服务器返回的报警输出参数（参阅六-71）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

94、int NETDVR_clearAlarms(int Handle);

功能	清除报警
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

95 、 int NETDVR_getAlarmOutTime(int Handle, unsigned char out_id, weekday_t weekday, struct NETDVR_alarmSetTime_t *p_para);

功能	获取报警输出布防参数

参数	Handle：七-3 返回的 DVR HANDLE out_id：第 out_id 路报警输出 weekday：星期几（参阅六-62） p_para：接收服务器返回的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

96、int NETDVR_setAlarmOutTime(int Handle, unsigned char out_id, weekday_t weekday, const struct NETDVR_alarmSetTime_t *p_para);

功能	设置报警输出布防参数
参数	Handle：七-3 返回的 DVR HANDLE out_id：第 out_id 路报警输出 weekday：星期几（参阅六-62） p_para：新的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

97、int NETDVR_setManualRecordParams(int Handle, const struct NETDVR_manualRecordParam_t *p_para);

功能	设置手动录像参数
参数	Handle：七-3 返回的 DVR HANDLE p_para：新的手动录像参数（参阅六-74）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

98、int NETDVR_getManualRecordParams(int Handle, struct NETDVR_manualRecordParam_t *p_para);

功能	获取手动录像参数
参数	Handle：七-3 返回的 DVR HANDLE p_para：接收服务器返回的手动录像参数（参阅六-74）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

99、int NETDVR_setNetworkParams(int Handle, const struct NETDVR_networkParam_t *p_para);

功能	设置网络参数
参数	Handle：七-3 返回的 DVR HANDLE p_para：新的网络参数（参阅六-75）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

100、int NETDVR_getNetworkParams(int Handle, struct NETDVR_networkParam_t *p_para);

功能	获取网络参数
参数	Handle：七-3 返回的 DVR HANDLE p_para：接收服务器返回的网络参数（参阅六-75）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

101、int NETDVR_initUpdate(int Handle, const struct NETDVR_updateParam_t *p_update_para, unsigned short *p_update_port);

功能	激活服务器准备升级、初始化升级参数并返回服务升级端口
参数	Handle：七-3 返回的 DVR HANDLE p_update_para：升级参数（参阅六-76） p_update_port：返回服务器升级端口指针
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

102、int NETDVR_regCBUpdateProgress(int Handle, PFUN_PROGRESS_T p_cb_func, unsigned int dwContent);

功能	注册升级进度回调函数
参数	Handle：七-3 返回的 DVR HANDLE p_cb_func：升级进度回调函数（参阅六-15） dwContent：用户自定义回调参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	在七-103 前使用

103、int NETDVR_startUpdate(int Handle, unsigned short update_port, char *s_update_dir, char *s_update_filename);

功能	开始升级
参数	Handle：七-3 返回的 DVR HANDLE update_port：服务器升级端口 s_update_dir：升级文件所在目录 s_update_filename：升级文件名
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

104、int NETDVR_stopUpdate(int Handle);

功能	停止升级
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

105、int NETDVR_restoreFactorySettings(int Handle);

功能	恢复默认设置
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

106、int NETDVR_reboot(int Handle);

功能	重启设备
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

107、int NETDVR_shutdown(int Handle);

功能	关闭设备
参数	Handle：七-3 返回的 DVR HANDLE
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

108、int NETDVR_getSystemTime(int Handle, unsigned int *p_system_time);

功能	获取系统时间
参数	Handle：七-3 返回的 DVR HANDLE p_system_time：接收服务器返回的系统时间
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

109、int NETDVR_setSystemTime(int Handle, const unsigned int system_time);

功能	设置系统时间
参数	Handle：七-3 返回的 DVR HANDLE system_time：新的系统时间
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

110、int NETDVR_getMotionDection(int Handle, unsigned char chn, struct NETDVR_motionDetection_t *p_para);

功能	获取移动侦测参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道号 p_para：接收服务器返回的移动侦测参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

111、int NETDVR_setMotionDection(int Handle, unsigned char chn, const struct NETDVR_motionDetection_t *p_para);

功能	设置移动侦测参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道号 p_para：新的移动侦测参数
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

112 、 int NETDVR_getMotionDectionTime(int Handle, unsigned char chn, weekday_t weekday, struct NETDVR_alarmSetTime_t *p_para);

功能	获取移动侦测布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 weekday：星期几（参阅六-62） p_para：接收服务器返回的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

113、int NETDVR_setMotionDectionTime(int Handle, unsigned char chn, weekday_t weekday, const struct NETDVR_alarmSetTime_t *p_para);

功能	设置移动侦测布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 weekday：星期几（参阅六-62） p_para：新的布防参数（参阅六-68）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

114、int NETDVR_getMotionDetectionAlarm(int Handle, unsigned char chn, struct NETDVR_alarmHandler_t *p_para);

功能	获取移动侦测报警处理参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：接收服务器返回的报警处理参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

115、int NETDVR_setMotionDetectionAlarm(int Handle, unsigned char chn, const struct NETDVR_alarmHandler_t

*p_para);

功能	设置移动侦测报警处理参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：新的报警处理参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

116 、 int NETDVR_getVideoLostTime(int Handle, unsigned char chn, weekday_t weekday, struct NETDVR_alarmSetTime_t *p_para);

功能	获取视频丢失布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：接收服务器返回的布防参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

117 、 int NETDVR_setVideoLostTime(int Handle, unsigned char chn, weekday_t weekday, const struct NETDVR_alarmSetTime_t *p_para);

功能	设置视频丢失布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：新的布防参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

118、int NETDVR_getSHTime(int Handle, unsigned char chn, weekday_t weekday, struct NETDVR_alarmSetTime_t *p_para);

功能	获取遮挡布防参数
参数	Handle：七-3 返回的 DVR HANDLE chn：通道 p_para：接收服务器返回的布防参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

119 、 int NETDVR_setSHTime(int Handle, unsigned char chn, weekday_t weekday, const struct NETDVR_alarmSetTime_t *p_para);

功能	设置遮挡布防参数
参数	Handle：七-3 返回的 DVR HANDLE

	chn：通道 p_para：新的布防参数（参阅六-69）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

120、int NETDVR_getUserList(int Handle, struct NETDVR_userList_t *p_user_list);

功能	获取用户列表
参数	Handle：七-3 返回的 DVR HANDLE p_user_list：接收服务器返回的用户列表（参阅六-80）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

121、int NETDVR_clearUserList(struct NETDVR_userList_t *p_user_list);

功能	清除用户列表
参数	p_user_list：要清除的用户列表（参阅六-80）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

122、int NETDVR_addUser(int Handle, const struct NETDVR_userInfo_t *p_user);

功能	添加用户
参数	Handle：七-3 返回的 DVR HANDLE p_user：需添加用户资料及权限（参阅六-81）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

123、int NETDVR_editUser(int Handle, const struct NETDVR_userInfo_t *p_user);

功能	编辑用户
参数	Handle：七-3 返回的 DVR HANDLE p_user：需编辑用户资料及权限（参阅六-81）
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

124、int NETDVR_deleteUser(int Handle, const char *user_name);

功能	删除用户
参数	Handle：七-3 返回的 DVR HANDLE user_name：需删除的用户名
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

125、int NETDVR_mutePreViewAudio(int Handle, unsigned char rcv_chn, bool bMute);

功能	控制预览音频是否静音（建议开启语音对讲时使用此接口），但不影响预览音频码流的接收
参数	Handle：七-3 返回的 DVR HANDLE rcv_chn：预览音频的通道 bMute：TRUE 时静音，FALSE 非静音
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

129、int NETDVR_remoteGetHddInfo(int Handle, struct NETDVR_hddInfo_t *p_hddinfo);

功能	获得远程硬盘信息
参数	Handle：七-3 返回的 DVR HANDLE; p_hddinfo 远程硬盘信息结构
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

127、int NETDVR_remoteGetSysVerInfo(int Handle, struct NET_SysVerInfo_t *p_verinfo);

功能	获得远程系统版本信息
参数	Handle：七-3 返回的 DVR HANDLE; p_verinfo 远程系统版本结构
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

128、int NETDVR_sdkCheckout(int Handle, char* chkstring);

功能	SDK 校验接口，保护二次开发商
参数	Handle：七-3 返回的 DVR HANDLE; chkstring: SDK 校验字符串
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

129、int NETDVR_getAlarmState(int Handle, struct NETDVR_AlarmIOState_t *pAlarmIOState);

功能	获得报警状态
参数	Handle：七-3 返回的 DVR HANDLE; pAlarmIOState: 报警输入输出状态
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

130、int NETDVR_openSubVideoReciever(int Handle, const struct NETDVR_subVideoParam_t *p_rcv_para, pFrameCallBack pCBFun, unsigned int dwContent);

功能	设置子码流接收
参数	Handle：七-3 返回的 DVR HANDLE; p_rcv_para: 子码流参数;,

	pCBFun:获得编码码流的回调函数, dwContent:回调上下文
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

131、int NETDVR_closeSubVideoReciever(int Handle, unsigned char rcv_chn);

功能	关闭制定通道的子码流接收
参数	Handle：七-3 返回的 DVR HANDLE; rcv_chn :接收的通道
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

132、int NETDVR_setSubDecoderFMT(int Handle, unsigned char rcv_chn, fmt_type_t fmt);

功能	设置子码流解码格式
参数	Handle：七-3 返回的 DVR HANDLE; rcv_chn :接收的通道 fmt 解码之后的格式
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

133、int NETDVR_createSubVideoDecoder(int Handle, unsigned char rcv_chn, pDecFrameCallBack pCBFun, unsigned int dwContent);

功能	创建子码流解码器
参数	参考主码流创建解码器函数: NETDVR_createVideoDecoder
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

134、int NETDVR_destroySubVideoDecoder(int Handle, unsigned char rcv_chn);

功能	销毁子码流解码器
参数	参考主码流创建解码器函数: NETDVR_destroyVideoDecoder
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

135、int NETDVR_startSubVideoSend(int Handle, const struct NETDVR_subVideoParam_t *p_rcv_para);

功能	开始发送子码流
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

136、int NETDVR_stopSubVideoSend(int Handle, unsigned char rcv_chn);

功能	停止发送子码流
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

137、int NETDVR_startSubRecord(int Handle, unsigned char chn, char *p_dir_path, unsigned int file_max_len);

功能	开始录制子码流码流

参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

138、int NETDVR_stopSubRecord(int Handle, unsigned char chn);

功能	停止录制子码流码流
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

139、int NETDVR_setSubRecordCB(int Handle, unsigned char chn, pFrameCallBack pRecordCBFun, unsigned int dwContent);

功能	设置子码流码流录制回调
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

140、int NETDVR_setSubRecordFileNameCB(int Handle, unsigned char chn, pRecFilenameCallBack pRecFilenameCBFunc, unsigned int dwContent);

功能	设置录像文件名回调
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

141、int NETDVR_Subsnapshot(int Handle, int chn, char *path, char *filename);

功能	抓图(子码流)
参数	
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

142、int NETDVR_setConnectTimeout(unsigned int millisecond);

功能	设置当前连接超时时间
参数	millisecond:更新的时间,单位毫秒
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略

143、int NETDVR_getConnectTimeout(unsigned int *pMillisecond);

功能	获得当前连接超时时间
参数	pMillisecond:获得超市时间,单位毫秒
返回	NETDVR_SUCCESS 成功；其他值失败，请查阅 NETDVR_RETURN_CODE。
前提	略